Exercise 3

2024-04-02

This code sets up the R environment with essential packages for data manipulation (tidyverse, lubridate), gender and race estimation (babynames, wru), data storage (arrow), and network analysis (igraph), enabling comprehensive data preparation and analysis

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.2     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.3     ✔ tibble    3.2.1
## ✔ lubridate 1.9.2     ✔ tidyr     1.3.0
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflic
ts to become errors
```

```
library(babynames)
```

```
## Warning: package 'babynames' was built under R version 4.3.3
```

```
library(wru)
```

```
## Warning: package 'wru' was built under R version 4.3.3
```

```
##
## Please cite as:
##
## Khanna K, Bertelsen B, Olivella S, Rosenman E, Rossell Hayes A, Imai K
## (2024). _wru: Who are You? Bayesian Prediction of Racial Category Using
## Surname, First Name, Middle Name, and Geolocation_. R package version
## 3.0.1, <https://CRAN.R-project.org/package=wru>.
##
## Note that wru 2.0.0 uses 2020 census data by default.
## Use the argument `year = "2010"`, to replicate analyses produced with earlier pack
age versions.
```

```
library(lubridate)
library(arrow)
```

```
## Warning: package 'arrow' was built under R version 4.3.3
```

```
##
## Attaching package: 'arrow'
##
## The following object is masked from 'package:lubridate':
##
##     duration
##
## The following object is masked from 'package:utils':
##
##     timestamp
```

```
library(igraph)
```

```
## Warning: package 'igraph' was built under R version 4.3.3
```

```
##
## Attaching package: 'igraph'
##
## The following objects are masked from 'package:lubridate':
##
##     %--%, union
##
## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union
##
## The following objects are masked from 'package:purrr':
##
##     compose, simplify
##
## The following object is masked from 'package:tidyr':
##
##     crossing
##
## The following object is masked from 'package:tibble':
##
##     as_data_frame
##
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
##
## The following object is masked from 'package:base':
##
##     union
```

Loading all the necessary files

```
setwd("E:\\McGill MMA\\Notes\\Winter 2 2024\\ORGB 672\\Assignments\\Exercise 3")
data_path <- "./"
app_data_sample <- read_parquet(paste0(data_path, "app_data_sample.parquet"))
edges <- read_csv(paste0(data_path, "edges_sample.csv"))
```

```
## Rows: 32906 Columns: 4
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## chr  (1): application_number
## dbl  (2): ego_examiner_id, alter_examiner_id
## date (1): advice_date
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

The code duplicates first names, estimates gender using historical baby names data, and predicts racial demographics from surnames. It also calculates examiners' tenure from application dates, enriching the dataset for in-depth demographic analysis of selected workgroups.

```
## Gender Estimation
# Get a list of first names without repetitions from app_data_sample
examiner_names <- app_data_sample %>%
  distinct(examiner_name_first) %>%
  mutate(examiner_name_first = tolower(examiner_name_first))

# Join with babynames dataset to get gender
examiner_names_gender <- examiner_names %>%
  inner_join(babynames %>% mutate(name = tolower(name)), by = c("examiner_name_first"
= "name")) %>%
  group_by(examiner_name_first) %>%
  summarize(
    gender = ifelse(sum(sex == "F") > sum(sex == "M"), "Female", "Male"),
    proportion_female = sum(sex == "F") / sum(sex %in% c("F", "M"))
  ) %>%
  ungroup()

# Join gender back to the app_data_sample
app_data_sample <- app_data_sample %>%
  mutate(examiner_name_first = tolower(examiner_name_first)) %>%
  left_join(examiner_names_gender, by = "examiner_name_first")

## Race Estimation
# Select and distinct last names for race estimation
examiner_surnames <- app_data_sample %>%
  select(surname = examiner_name_last) %>%
  distinct()

# Predict race based on surnames using wru
examiner_race <- predict_race(voter.file = examiner_surnames, surname.only = TRUE) %
>%
  as_tibble()
```

```
## Predicting race for 2020
```

```
## Warning: Unknown or uninitialised column: `state`.
```

```
## Proceeding with last name predictions...
```

```
## ℹ All local files already up-to-date!
```

```
## 701 (18.4%) individuals' last names were not matched.
```

```
# Determine the race with the highest probability
examiner_race <- examiner_race %>%
  mutate(max_race_p = pmax(pred.asi, pred.bla, pred.his, pred.oth, pred.whi)) %>%
  mutate(race = case_when(
    max_race_p == pred.asi ~ "Asian",
    max_race_p == pred.bla ~ "black",
    max_race_p == pred.his ~ "Hispanic",
    max_race_p == pred.oth ~ "other",
    max_race_p == pred.whi ~ "white",
    TRUE ~ NA_character_
  ))

# Join race data back to the app_data_sample
app_data_sample <- app_data_sample %>%
  left_join(examiner_race, by = c("examiner_name_last" = "surname"))

## Tenure Calculation
# Ensure dates are in the correct format and calculate tenure
examiner_dates <- app_data_sample %>%
  select(examiner_id, filing_date, appl_status_date) %>%
  mutate(
    start_date = ymd(filing_date),
    end_date = as_date(dmy_hms(appl_status_date))
  )

# Calculate tenure for each examiner
examiner_dates <- examiner_dates %>%
  group_by(examiner_id) %>%
  summarise(
    earliest_date = min(start_date, na.rm = TRUE),
    latest_date = max(end_date, na.rm = TRUE),
    tenure_days = interval(earliest_date, latest_date) %/% days(1)
  ) %>%
  filter(year(latest_date) < 2018)

# Join tenure data back to the app_data_sample
app_data_sample <- app_data_sample %>%
  left_join(examiner_dates, by = "examiner_id")

# Display the head of the final enriched dataset
head(app_data_sample)
```

```
## # A tibble: 6 × 28
##   application_number filing_date examiner_name_last examiner_name_first
##   <chr>              <date>      <chr>              <chr>
## 1 08284457           2000-01-26  HOWARD             jacqueline
## 2 08413193           2000-10-11  YILDIRIM           bekir
## 3 08531853           2000-05-17  HAMILTON           cynthia
## 4 08637752           2001-07-20  MOSHER             mary
## 5 08682726           2000-04-10  BARR               michael
## 6 08687412           2000-04-28  GRAY               linda
## # ℹ 24 more variables: examiner_name_middle <chr>, examiner_id <dbl>,
## #   examiner_art_unit <dbl>, uspc_class <chr>, uspc_subclass <chr>,
## #   patent_number <chr>, patent_issue_date <date>, abandon_date <date>,
## #   disposal_type <chr>, appl_status_code <dbl>, appl_status_date <chr>,
## #   tc <dbl>, gender <chr>, proportion_female <dbl>, pred.whi <dbl>,
## #   pred.bla <dbl>, pred.his <dbl>, pred.asi <dbl>, pred.oth <dbl>,
## #   max_race_p <dbl>, race <chr>, earliest_date <date>, latest_date <date>, …
```

This code isolates data for specific workgroups identified by art unit prefixes “165” or “172”, cleanses it by selecting relevant columns and omitting incomplete records, preparing the subset for focused demographic and network analysis.

```
# Filter workgroups with art units starting with "165" or "172"
filtered_workgroups <- app_data_sample %>%
  filter(substr(examiner_art_unit, 1, 3) == "165" | substr(examiner_art_unit, 1, 3) =
= "172")

# Select necessary columns and remove rows with NAs in those columns
cleaned_filtered_workgroups <- filtered_workgroups %>%
  select(application_number, examiner_name_last, examiner_name_first, gender, race, t
enure_days, examiner_art_unit) %>%
  na.omit()

# Display the dimensions of the cleaned and filtered workgroups
print(dim(cleaned_filtered_workgroups))
```

```
## [1] 138165      7
```

This step converts the examiner_art_unit to a string to facilitate text operations, then filters the dataset to focus on two specific examiner workgroups, enabling targeted analysis of their network structures and demographics.

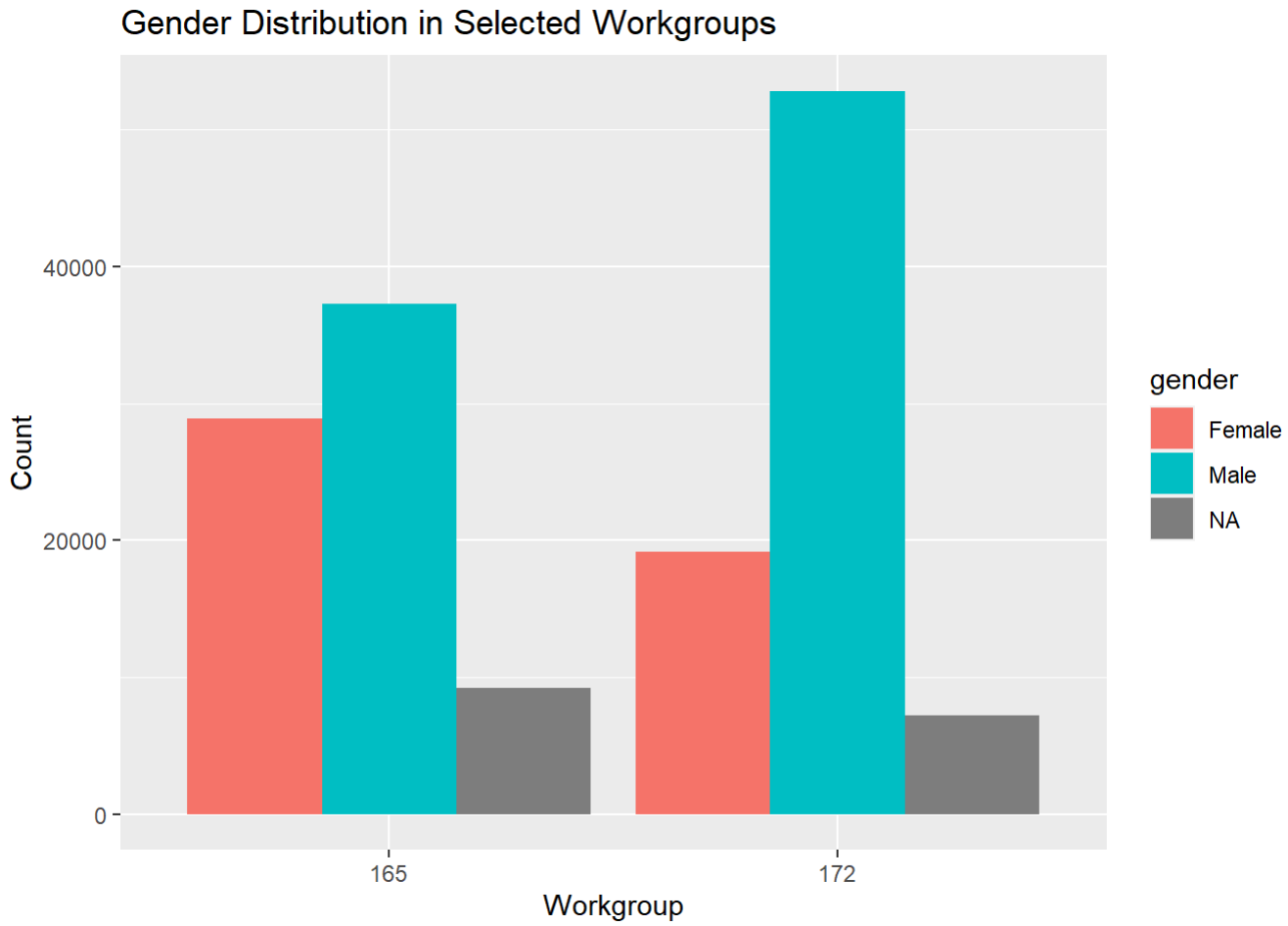
```
## Step 1: Filter for Selected Workgroups
# Convert examiner_art_unit to character to ensure string operations work correctly
app_data_sample <- app_data_sample %>%
  mutate(examiner_art_unit = as.character(examiner_art_unit))

# Filter for workgroups 165 and 172
selected_workgroups <- app_data_sample %>%
  filter(str_starts(examiner_art_unit, "165") | str_starts(examiner_art_unit, "172"))
```

The code calculates summary statistics and generates visualizations to compare gender, race, and tenure across the selected workgroups, providing a quantitative and visual baseline for understanding their demographic composition.

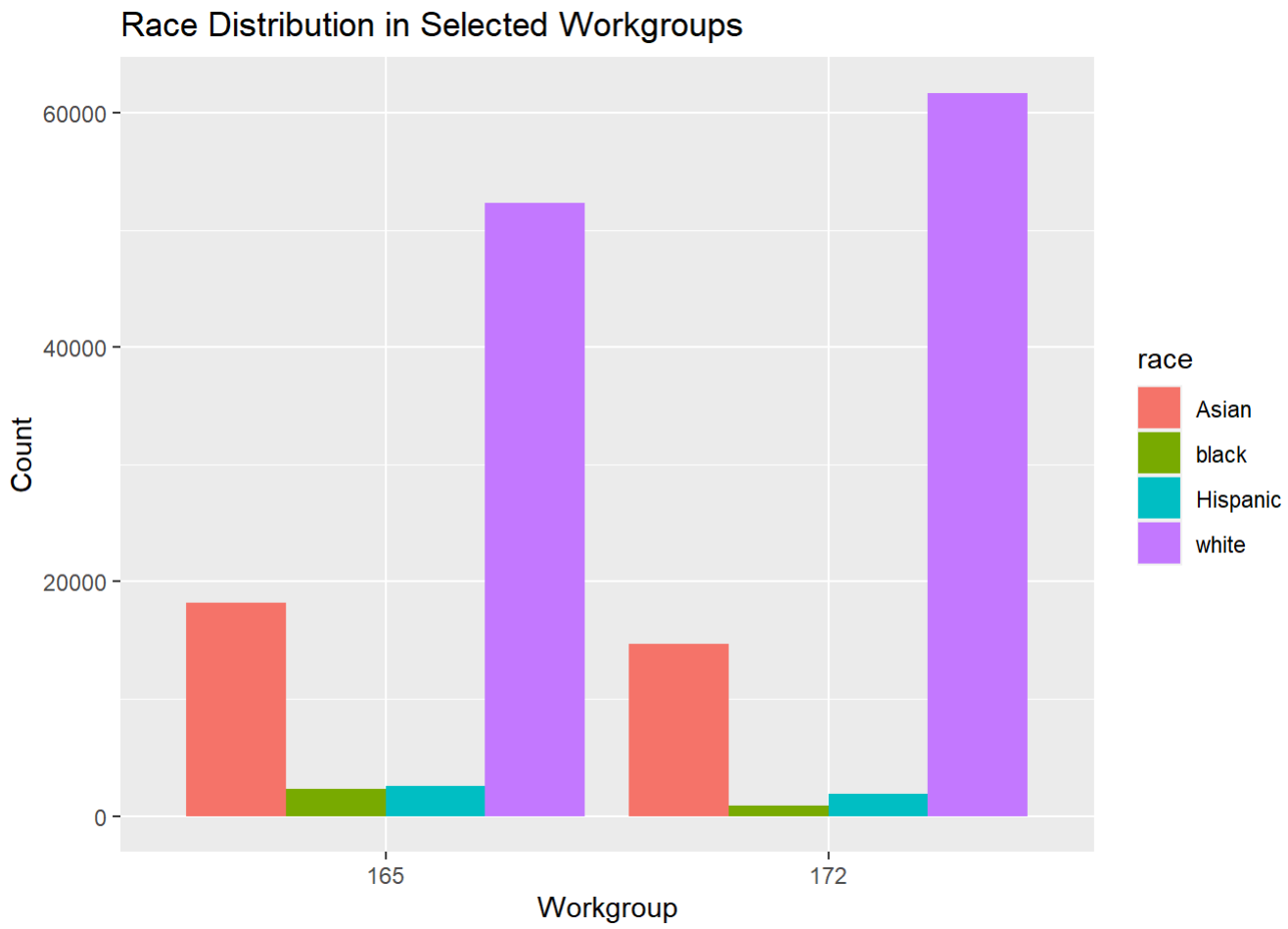
```
## Step 2: Summary Statistics and Plots
# Summary statistics for gender, race, and tenure
summary_stats <- selected_workgroups %>%
  group_by(substring(examiner_art_unit, 1, 3), gender, race) %>%
  summarise(
    avg_tenure_days = mean(tenure_days, na.rm = TRUE),
    n = n(),
    .groups = 'drop'
  )

print(summary_stats)
```

```
## # A tibble: 22 × 5
##    `substring(examiner_art_unit, 1, 3)` gender race     avg_tenure_days     n
##    <chr>                                <chr>  <chr>              <dbl> <int>
##  1 165                                  Female Asian              4974.  2937
##  2 165                                  Female Hispanic           6064.  2081
##  3 165                                  Female white              5895. 23906
##  4 165                                  Male   Asian              5904. 10281
##  5 165                                  Male   Hispanic           4992.   494
##  6 165                                  Male   black              6221.  1587
##  7 165                                  Male   white              5769. 24884
##  8 165                                  <NA>   Asian              5883.  4952
##  9 165                                  <NA>   black              6137    705
## 10 165                                  <NA>   white              6196.  3563
## # ℹ 12 more rows
```

```
# Plots for demographics
# Gender distribution
gender_dist_plot <- ggplot(selected_workgroups, aes(x = substring(examiner_art_unit,
1, 3), fill = gender)) +
  geom_bar(position = "dodge") +
  labs(title = "Gender Distribution in Selected Workgroups", x = "Workgroup", y = "Co
unt")

# Race distribution
race_dist_plot <- ggplot(selected_workgroups, aes(x = substring(examiner_art_unit, 1,
3), fill = race)) +
  geom_bar(position = "dodge") +
  labs(title = "Race Distribution in Selected Workgroups", x = "Workgroup", y = "Coun
t")

print(gender_dist_plot)
```




```
print(race_dist_plot)
```



This code harmonizes data types to merge examiner records, constructs advice networks for selected workgroups, and computes centrality measures, establishing a foundation for analyzing the influence and connectivity of examiners within these networks.

```
## Step 3: Create Advice Networks and Calculate Centrality Scores
# Convert examiner_id in edges to character to match types
edges <- edges %>%
  mutate(ego_examiner_id = as.character(ego_examiner_id),
         alter_examiner_id = as.character(alter_examiner_id))

# Convert examiner_id in selected_workgroups to character to match the types in edges
selected_workgroups <- selected_workgroups %>%
  mutate(examiner_id = as.character(examiner_id))

# Now perform the join with matching types
selected_edges <- edges %>%
  inner_join(selected_workgroups %>% select(examiner_id), by = c("ego_examiner_id" =
"examiner_id")) %>%
  select(ego_examiner_id, alter_examiner_id)
```

```
## Warning in inner_join(., selected_workgroups %>% select(examiner_id), by = c(ego_e
xaminer_id = "examiner_id")): Detected an unexpected many-to-many relationship betwee
n `x` and `y`.
## ℹ Row 1 of `x` matches multiple rows in `y`.
## ℹ Row 70 of `y` matches multiple rows in `x`.
## ℹ If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
# Filter edges for selected workgroups by joining with the selected workgroups
# Assuming that examiner_id is already a character in selected_workgroups
selected_edges <- edges %>%
  inner_join(selected_workgroups %>% select(examiner_id), by = c("ego_examiner_id" =
"examiner_id")) %>%
  select(ego_examiner_id, alter_examiner_id)
```

```
## Warning in inner_join(., selected_workgroups %>% select(examiner_id), by = c(ego_e
xaminer_id = "examiner_id")): Detected an unexpected many-to-many relationship betwee
n `x` and `y`.
## ℹ Row 1 of `x` matches multiple rows in `y`.
## ℹ Row 70 of `y` matches multiple rows in `x`.
## ℹ If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
# Create advice networks
g <- graph_from_data_frame(selected_edges, directed = TRUE)
```

```
## Warning in graph_from_data_frame(selected_edges, directed = TRUE): In `d' `NA'
## elements were replaced with string "NA"
```

```
# Calculate centrality scores (e.g., degree centrality)
degree_centrality <- degree(g, mode = "in")
betweenness_centrality <- betweenness(g)

# Associate centrality scores with examiners
centrality_scores <- data.frame(
  examiner_id = V(g)$name,
  degree = degree_centrality,
  betweenness = betweenness_centrality
)

# Make sure examiner_id is a character in both data frames before joining
selected_workgroups <- selected_workgroups %>%
  mutate(examiner_id = as.character(examiner_id))

centrality_scores <- centrality_scores %>%
  mutate(examiner_id = as.character(examiner_id))
```

This step integrates centrality measures with examiner demographics to explore relationships, exemplified by correlating tenure with degree centrality, facilitating insights into how network position correlates with examiner experience.

```
## Step 4: Analyze Relationship Between Centrality and Examiner Demographics
# Merge centrality scores with demographic data
analysis_data <- selected_workgroups %>%
  inner_join(centrality_scores, by = "examiner_id")

# Example analysis: Correlation between tenure and centrality
cor_analysis <- cor.test(analysis_data$tenure_days, analysis_data$degree, use = "comp
lete.obs")

print(cor_analysis)
```

```
##
##  Pearson's product-moment correlation
##
## data:  analysis_data$tenure_days and analysis_data$degree
## t = 41.522, df = 82074, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1367294 0.1501306
## sample estimates:
##       cor
## 0.1434366
```

The gender distribution graph reveals a male-dominated workgroup 172, contrasting with a more gender-balanced workgroup 165. Racially, both workgroups are largely white, with 172 showing greater racial diversity. The correlation result (r = 0.143) implies that longer tenure moderately correlates with higher centrality in the advice network, indicating that examiners with more service years may hold more central advisory roles, although other factors also influence this relationship.